Programa del curso

Semestre 2019-20

Nombre del curso:	Construcción de Aplicaciones Móviles
Course Name:	Mobile App Development
Créditos:	3
Profesor:	Mario Linares (m.linaresv@uniandes.edu.co).
Versión PDF	Click Aquí

Descripción

The main goal of this course is to promote/generate in the students, the skills required to conceive, design, build, and test software solutions based on mobile platforms, by following a development process that considers mobile app development-specific challenges and design constraints. Therefore, during the course, the students will explore concepts related not only to mobile devices, but also infrastructures, environments and software lifecycles.

This course is taught from the perspective of Software Engineering, Design, and Architecture, having as main-drivers the design and implementation of software solutions considering design constraints and in particular three quality attributes: performance, eventual connectivity, and accessibility. As the main deliverable of the course, the students have to implement a real product (i.e., software solution that provides value) following an iterative and incremental process; the iterations in the process are aligned with the course modules in such a way that the product from each iteration has to include the concepts presented/explored in the corresponding module. **Note that the course focuses on design, and software engineering and architecture. This is not a programming course.**

The mobile development process performed by the students includes the conception, design, programming, and testing phases, and aims at building a software solution that includes apps deployed in mobile devices by using two of three dominant platforms in the market: Android-Java1, Android-Kotlin & iOS2. To provide the students with the knowledge of how to use the mobile development APIs and platforms, we provide the students with lists of resources they can check to learn mobile programming concepts by their own.

It is worth noting that at the end of the course, the students should release a functional

app that includes all the concepts presented in the lectures and labs. In addition, this course will explore architectures and internals of the mobile operative systems, which is highly relevant and pertinent content to understand several performance and security issues in mobile apps.

At the end of the course, you should be able to:

- **O1:** Design mobile apps by considering design constraints and in particular three quality attributes: performance, eventual connectivity, and accessibility
- **O2:** Build native mobile apps that provide users with a multi-device experience, and use basic programming components, sensors, location services, and data storage.
- **O3:** Justify design and implementation decisions based on the specifics/particularities in (i) the mobile ecosystem, (ii) the development process, and (iii) the dominant mobile platforms.
- **O4:** Use tools that support software development tasks such as testing, resources profiling/monitoring, and vulnerabilities detection.

[tab title="Metodology"]

Metodology

This course syncretizes different methodologies such as flipped learning/classroom, and challenges-based learning, and incremental learning but with a special focus on project-based learning. A project is the main component that serves as the root and final goal of every activity in the course. This course content is organized in 4 teaching modules that present/explore specific topics in an incremental and complementary way. Each one of the modules includes activities oriented to practice the learned concepts and to develop the project iteratively. Therefore, a learning module is also an iteration/sprint.

A sprint represents a clear challenge that working teams must achieve; each challenge imposes specific design constraints and service level agreements that are directly related to the corresponding learning module. A challenge is also an iteration in the pure agile philosophy, with a defined time window, and the expectation of a complete product to be delivered. Each sprint is divided into weekly micro-sprints that are designed to help the teams to accomplish a challenge.

Note that the micro-sprints start and end on Friday, and the teams must deliver tasks at the end of each micro-sprint mostly (i.e., on Friday midnight). On purpose, the lab sessions will be on Friday in order to have a micro-sprint closing moment with different learning activities (e.g., quizzes, cross-validations, etc.). Because we follow and incremental model, the products of each micro-sprint are small steps for building the product expected to have for the corresponding sprint.

In summary, each sprint can be defined as a tuple S = <challenge, learning module, micro-sprints>.

Concerning the relation between the challenges and flip-flop learning, the course design carefully defines a set of activities that emphasizes student autonomy and discipline. There will be outside-class activities that must be finished by the students before each class; examples of these outside-class activities are on-the-field data collection tasks, and reading of our online book (yes, we have an online book). Remember that this is also a challenge-oriented course in which solutions to the challenges are designed by the students. This is shift from traditional lectures the students are used to. The classes will be devoted to team work and feedback sessions, because the students must prepare the class a-priori and be ready to discuss the class content. We will have few traditional lectures because some topics are more suitable for a traditional lecture. Anyway, the classes are expected to be more interesting, dynamic, interactive, and focused on the specific issues the students experience with the outside-class activities.

This class requires from the students, discipline and compromise with the class activities.

[tab title="Content and Modules"]

Content and Modules

This is not a programming course. This is a software engineering and architecture course. **Therefore, we will not teach you the language syntaxes or the specifics of each language.** There are courses and books specifically designed to teach you how to program Android, Kotlin, or iOS apps, and we encourage you to go there if you want to be a specialist programmer.

As mentioned before, this course content is organized in 4 teaching modules that present/explore specific topics in an incremental and complementary way. Each one of the modules includes activities oriented to practice the learned concepts. These activities are expected to help the student to explore/learn in a guided and autonomous way the main concepts and topics regarding mobile app development

The content modules are described as follows:

***Module 1 - Context and Conception (August 6th September 7th):**

- Architecture concepts recap.
- Design thinking philosophy.
- The mobile ecosystem and types of mobile apps
- Mobile App life cycle and app conception
- UX/UI in mobile apps; GUI metaphors and components

Module 2 - Design and Eventual connectivity (September 8th October 5th):

- Architecture of the Mobile Operating Systems.
- Programming model and components: Android, Kotlin, iOS.

- Architectures and patterns for mobile applications.
- Local and external storage. Best practices for eventual connectivity management.

*** Module 3 - Performance, and Services (October 6th November 2nd):**

- Policies (OSes and Frameworks) that impact mobile app performance.
- Asynchronous services, threads, backend services, and background services.
- Best practices for performance.
- Analytics/ML-based services.
- Tools: profiling, monitoring, linting

*** Module 4 - Security and Testing (November 3rd November 23rd):**

- Vulnerabilities in the mobile operating systems.
- Malware and vulnerabilities in mobile apps.
- Tools for vulnerability and malware detection.
- Challenges, types, and approaches for mobile app testing.
- Tools for mobile app testing: fuzzers, rippers, cloud-based services, automation APIS

[tab title="Grading"]

Grading

- **The grading/evaluation scheme is the following:**

Grading scheme		Percentaje
Midterm and Exam (45%)	**Midterm**	20%
	Exam	25%
Project (40%)	Challenge 1	10%
	Challenge 2	10%
	Challenge 3	10%
	Challenge 4	10%
Micro-sprints activities (15%		15%

- **One mid-term + One exam**
- **Project:** it includes 4 development iterations aligned with the 4 teaching modules; therefore, each iteration focuses on different quality attributes. A project is developed by teams composed of 4 students. At the end of each iteration (except for the initial iteration that is devoted to conceiving the app idea) a working product must be delivered in two of the three platforms (Android, Kotlin, iOS). The students will follow a rotating-leader scheme, i.e., each iteration will have a different student leading the team.

- **Micro-sprint activities:** it refers to quizzes, lecture controls, class preparation/participation. In the weekly micro-sprint plan we will let you know what activities are graded. These activities are mandatory, thus, if one activity is not sent to the corresponding grader, then the grade (for the activity) will be 0/5.
- **Micro-challenges:** additional tasks will be suggested during the semester. Those tasks are not mandatory but will have extra points over the final grade.

The grade for a project iteration is computed individually. The delivered product, the process, auto-assessment, and peer-assessments are components of each iteration grade. The quality of the product and process is evaluated based on (i) the artifacts (i.e., repository, documentation, etc.), (ii) the expected features, requirements and constraints of each iteration, (iii) individual contributions, and (iv) individual presentation of the product. The product (at the end of each iteration) is expected to work properly on real devices.

Other grading guidelines

- Grades (for midterms, iterations, etc.) are in the range [0.00, 5.00] with two decimal figures
- Final grades will be in the range [1.50, 5.00] with two decimal figures precision. In the cases of grades with more than two decimal figures, classic rounding will be applied to the decimal figures in order to obtain two decimal figures.
- This course is approved with a final grade of 3.00 / 5.00. There is no automatic approximation, e.g., there is no approximation from 2.99 to 3.00.

[tab title="Resources"]

Resources

The course activities will be supported on the following infrastructure:

- An online virtual classroom at SICUA plus (http://sicuaplus.uniandes.edu.co) that contains the schedule, activities, micro-sprint plans, challenges, etc.
- An online gitbook with content that students must read and work with as part of the outside-class activities
- The teams will use Git as the version control system (VCS) and GitLab (https://about.gitlab.com/) for the repository management. Each team is responsible for creating its private repository and giving Reporter privileges to the professor and lab instructors. The teams must use the GitLab features for project management, bug/issue tracking, continuous integration, wiki, and pages.
- The Department has development environments (PC-based and Mac) in its development labs: Wuaira and Turing labs.
- Mobile devices such as phones, tablets, and smart watches are available under request. In order to use the devices, the students should accept the usage policy and request the devices at http://labmoviles.virtual.uniandes.edu.co. The students enrolled in the course will be registered in the system during the second week of

classes. Any violation to the "usage policy" will drive the student to be banned in the system; any student requesting a device will be responsible for any damage in the device.

- The apps can be developed/tested using devices different to the ones owned by the Department (i.e., the ones in the "mobile computation lab"), but with the following conditions:
    - The devices should have the minimum specifications required by the course activities.
    - The availability of the device must not be an excuse for not finishing (on time) any if of course activities/deliverables.
    - The student is responsible for any technical support required for the device.