**CS673 Software Engineering** 

**Team 3 - Project Name**

**Project Proposal and Planning**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Magnus Urosev | Team Leader / Partial Design and Implementation Leader (Stats Models) | *Magnus Urosev* | 5/13/24 |
| Adrian Ortiz | Configuration Leader / Security Leader |  |  |
| Xi Zeng | QA Leader |  |  |
| Jack Cairns |  |  |  |
| Jianing Li? | ? |  |  |
| Open Roles | Requirement Leader,  Design and Implementation Leader |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| **v0.01** | **Magnus** | **5/12** | **Added draft Overview and Related Work** |
| **v0.02** | **Magnus** | **5/13** | **Expanded more on project details and timeline** |
|  |  |  |  |
|  |  |  |  |

[Overview](#_heading=h.gjdgxs)

[Related Work](#_heading=h.30j0zll)

[Proposed High level Requirements](#_heading=h.1fob9te)

[Management Plan](#_heading=h.3znysh7)

[Objectives and Priorities](#_heading=h.2et92p0)

[Risk Management (need to be updated constantly)](#_heading=h.tyjcwt)

[Timeline (need to be updated at the end of each iteration)](#_heading=h.3dy6vkm)

[Configuration Management Plan](#_heading=h.1t3h5sf)

[Tools](#_heading=h.4d34og8)

[Deployment Plan if applicable](#_heading=h.2s8eyo1)

[Quality Assurance Plan](#_heading=h.17dp8vu)

[Metrics](#_heading=h.3rdcrjn)

[Code Review Process](#_heading=h.26in1rg)

[Testing](#_heading=h.lnxbz9)

[Defect Management](#_heading=h.35nkun2)

[References](#_heading=h.1ksv4uv)

[Glossary](#_heading=h.44sinio)

# Overview

(Please give an overview of your project. It should include the motivation, the purpose and the potential users of the proposed software system, the basic functionality of the proposed software system and the possible technology stack to be used. )

* Assessment and analytics service: progress tracking, feedback, reports management, etc. This service allows for psychometric analysis of examinee performance using a Rasch Framework. It can be used to report back to the student their deficiencies in certain content areas/domains. Professors or classroom facilitators will also be able to receive statistics of examinees/classroom such as their theta estimates and SEMs, raw and scaled scores.
  + Support of IRT Analysis and input of IRT parameters and Rasch Model
  + Python, Django and REST, PyIRT, Pandas, NumPy, Celery RabbitMQ or Redis, Matplotlib

# Related Work

(Please describe any similar software systems that you have found through the online research, and the differences between your software and those software systems.)

* Blackboard, Coursera, Udemy, SAT and ACT, PearsonVUE Exam Developer

# Proposed High level Requirements

* 1. Functional Requirements  
     (For each functional requirement, please give a feature title and a brief description using the following format: As (a role), I want to (action), so that (value).)
     1. Essential Features (the core features that you definitely need to finish):

(For each essential features, please give a rough estimation in terms of person hours or an range of person hours)

Desirable Features (the nice features that you really want to have too):

* + 1. Optional Features (additional cool features that you want to have if there is time):
    2. Existing Features (delete this item if your project starts from scratch):
  1. Nonfunctional Requirements
     1. Security requirements

# Management Plan

## Objectives and Priorities

(Please describe your project objectives with highest priority first. Project Goals can include but not limited to complete all proposed (essential) features, deploy the software successfully, the software has no known bugs, maintain high quality, etc )

* Be able to create exam items (all active) that have a difficulty measure (IRTb), supports Rasch (all discriminations are constant), support only fixed-form for now, allows creation and testing of new forms, return CI of theta estimate, provide Raw and Scaled score to professor or class facilitator
* Be able to pull exam and performance information from front-end teams and do analysis on backend then provide display/data to FE team

## Risk Management (need to be updated constantly)

(Please write a summary paragraph about the main risks your group identified and how you plan to manage these risks. Then use the separate google sheet for detailed risk management. The template is provided in the same folder with this file. Please provide the link to the sheet.)

**Risk Management Sheet Link:**

* Unable to create model due to depreciating library
  + Could switch to R or manually create model from scratch (Magnus)
* Small group size due to dropped class
* Overlapping services with another team

## Timeline (this section should be filled in iteration 0 and updated at the end of each later iteration)

| Iteration | Functional Requirements(Essential/Disable/Option) | Tasks (Cross requirements tasks) | Estimated/real person hours |
| --- | --- | --- | --- |
| 1 | Create IRT model and Test Cases |  |  |
| 2 | Receive data from FE |  |  |
| 3 | Display data and information |  |  |

# Configuration Management Plan

## Tools

(In this project, we will use Git and Github as the version control tools. Please also specify any other tools to be used, e.g. IDE tools, CI/CD tools, container tools, SAST or DAST tools, and any other DevOps tools)

* 1. Code Commit Guideline and Git Branching Strategy  
     (Please briefly describe criteria for the code commitment and the branching strategy used, e.g. what are the branches to be used, how the pull request will be used etc. Here is an article to give you some basic knowledge about different git branching strategies: <https://www.flagship.io/git-branching-strategies/>

## Deployment Plan if applicable

(If you plan to deploy your application (e.g. your web application), briefly describe how you plan to deploy your application).

# Quality Assurance Plan

## Metrics

(Describe the metrics to be used in the project to measure the quality of your software. Each metric should be measurable and quantifiable. Examples of metrics include product complexity (KLOC, # of files, # of classes, # methods, cyclomatic complexity, etc.) , defect rate (# of defect per KLOC), # of test cases, test case pass rate, cost (# of person hours used), # of user stories completed, etc. **The result of these metrics should be reported in the progress report/ iteration summary sheet.**)

| Metric Name | Description |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* 1. Coding Standard

(Describe any coding standard to be used)

* PEP8 for Python
* APA 7th ed. Styling for any Statistics or Math output

## Code Review Process

(Everyone should review all documents to be submitted. Here you will mainly describe how the code review will be done. Who will review the code, e.g. design or implementation leader will review all code or team members review each other’s code. Do you use pull requests for the code review? Is there a checklist to help review? What feedback should the reviewer provide?)

## Testing

(Both manual testing and automated testing should be considered. Both unit testing and integration testing should be considered. Briefly describe the testing tools/framework to be used, the personnel involved (e.g. the QA leader will focus on the integration testing and each developer will unit test their own code), when and what types of testing will be performed, the testing objectives, etc)

## Defect Management

(Describe the tool to be used to manage the defect (e.g github issues). The types of defects to look at. The actions or personnel for defect management. )

# References

(For more details, please refer to the encounter example in the book or the software version of the documents posted on blackboard. )

# Glossary

(Any acronym used in the document should be explained here)

* IRT - Item Response Theory
* Rasch - Rasch Modeling