# ANDROAI

## AI-Powered Automatic Android Program Builder

**Powered By [AndroAI Organizations]**

GEBZE
TECHNICAL UNIVERSITY

# AndroAI Project Summary and Plan

## 1. Project Summary

### 1.1. Project Objective:

The objective of the AndroAI project is to enable users to automatically generate Android applications based on specified requirements through a form-based web interface. This system uses an LLM (Large Language Model) to generate Android application codes according to the inputs provided by the user, compiles these codes, and provides the APK output to the user. It also features an automatic correction and retry mechanism in case of errors.

### 1.2. Project Scope:

The project includes a React-based web interface, a NestJS-based server, and a backend system integrated with LLM. Users specify their Android application requirements via the web interface, these requirements are sent to the LLM, and appropriate Android codes are generated. The codes are compiled on the server, and the APK output is provided to the user. The system also includes an automatic error correction and retry mechanism. Initially, the system will perform core functions, with more complex features to be added later.

### 1.3. Project Goals:

- Create a user-friendly web interface.
- Accurately and completely generate Android codes based on user requirements.
- Successfully compile the generated codes and provide the APK output. (With Gradle or or a makefile (via Javac))
- Automatically detect and correct errors and retry until successful. (In the first phase, it will be continued until we get an apk output.)
- Design the system to be modular and extendable. (This step is very important for prompt expansion and compatibility with other LLMs.)

### 1.4. Project Stakeholders:

- **AndroAI Organization:** The executing and managing organization of the project. (GTU stundents)
- **Project Team:** The individuals responsible for developing and maintaining the project.
- **Users:** The end-users who will use the system to create Android applications.

- **LLM Providers:** Organizations or services providing and supporting the LLM technology.

### 1.5. Project Team:

- **Ahmet Özdemir:** Project Manager and Backend Developer
- **Yasir Şekerci:** Backend Developer & DevOps
- **Musab Kardeş:** Full Stack Developer
- **Ömer Sarıçam:** Backend Developer
- **Muhammed Yasir Güneş:** Backend Developer & Servers Side Developer
- **Akif Safa Angi:** Backend Developer Engineer
- **Feridun Taha Açıkyürek:** Frontend Developer & Test and Quality Assurance Specialist

## 2. Project Details

### 2.1. Project Schedule:

1. **Project Planning:**
   - Requirements gathering, project planning and training
   - Estimated duration: 1-2 weeks

2. **User Interface Design and Development:**
   - Design and development of React based web interface
   - Creation of user forms
   - Estimated duration 1-2 weeks

3. **Backend and API Development:**
   - Development of NestJS based backend
   - Integration with LLM and API development
   - Estimated duration 3-4 weeks

4. **Code Generation and Compilation:**
   - Code generation using LLM
   - Compiling the generated codes and creating APK output
   - Estimated duration: 3-4 weeks

5. **Testing and Debugging:**
    - Testing the whole system
    - Error correction and retesting
    - Estimated duration: 3-4 weeks

6. **System Integration and Final Test:**
    - Integration of all modules
    - Final test
    - Estimated duration: 2-3 weeks

7. **Publishing and Distribution:**
    - First release of the system
    - Distribution to users
    - Estimated duration: 1-2 weeks

These steps will naturally be intertwined in some places. So the total time given to these steps is not the total time for the first release of the project.

## 2.2. Important Dates:

1. **Project Start Date:**
    - Date July 7, 2024

2. **Completion of Requirements and Planning:**
    - Date July 15, 2024

3. **Completion of UI Design and Development:**
    - Date July 25, 2024

4. **Completion of Backend and API Development:**
    - Date August 15, 2024

5. **Code Generation and Completion of Compilation:**
    - Date August 15, 2024

6. **Testing and Debugging Completion:**

   - Date August 22, 2024

7. **Completion of System Integration and Final Testing:**
   - Date August 30, 2024

8. **First Release and Distribution:**
   - Date September 3, 2024

We have estimated the dates here, these dates may vary depending on the problems that may arise. To minimize this possibility, we will keep the distribution of tasks updated.

## 2.3. Risk Management

### Potential Risks and Solutions:

1. **Technical Challenges:**
   - **Risk:** LLM may not generate correct and error-free code according to user requirements.
   - **Solution:** Continuously update and test LLM models. Effective use of debugging module for automatic error detection and correction. Use alternative AI models or train the AI model effectively via API.

2. **Project Planning:**
   - **Risk:** Failure to adhere to the set timeline.
   - **Solution:** Prepare a detailed project plan and review it regularly. Use Jira software for project management, including sprints and task assignments. Allocate additional resources and time as needed. Regularly monitor progress and make adjustments as needed to stay on track.

3. **Resource Insufficiency:**
   - **Risk:** Lack of sufficient human resources during the development process.
   - **Solution:** Maintain high motivation among volunteer developers and include new volunteers into the project when needed.

4. **Integration Issues:**
   - **Risk:** Integration issues between web interface, backend and LLM.
   - **Solution:** Establish a modular and testable structure. Implement continuous integration and continuous delivery (CI/CD) processes.

5. **User Feedback:**
    - **Risk:** Users may not adapt to the system or their expectations may not be met.
    - **Solution:** Design a user-friendly interface and regularly collect and evaluate user feedback.