# AImpact: Your AI co-founder. Whitepaper

Placeholder
OSTOLEX
placeholder@ostolex.com

June 3, 2025

**Abstract**

AImpact represents an AI-powered platform designed to democratize Web3 development by enabling users to create blockchain applications through natural language prompts. It eliminates traditional coding barriers, allowing users to describe their Web3 project ideas in plain language and receive fully functional applications in return. The platform's AI-driven approach reduces development time and complexity while maintaining the security and functionality expected from Web3 applications.

**Keywords:** artificial intelligence, Web3, blockchain, Solana, smart contracts, decentralized applications, AI-powered development, prompt-based programming, DApp generation

# Contents

# 1 Introduction

The rapid evolution of blockchain technology and Web3 applications has created unprecedented opportunities for innovation in decentralized systems. However, the technical complexity of blockchain development remains a significant barrier to entry for many potential innovators. Traditional Web3 development requires extensive knowledge of specialized programming languages, blockchain protocols, and cryptographic principles, often deterring creative individuals and businesses from participating in this transformative technology.

AImpact emerges as a solution to this challenge by leveraging artificial intelligence to bridge the gap between ideas and implementation. Our platform transforms natural language descriptions into fully functional Web3 applications, democratizing access to blockchain technology and enabling a new wave of innovation in the decentralized space.

This whitepaper outlines the technical architecture, capabilities, and potential applications of AImpact.

# 2 Userflow

The typical user flow for the AImpact platform consists of the following steps:

1. **Project Description**: Users provide a natural language description of their desired Web3 application, including functionality, features, and requirements.

2. **Application Design Generation**: The platform's AI system generates a user interface for the application. User can continiously modify the design of the application with text prompts or images.

   *Note: user can choose to skip this step and start developing frontend code without a design. This is useful for quick prototyping and allow user to make changes to the design more easily.*

3. **Frontend Code Generation**: Based on the provided design, the AI generates the frontend code for the application. Frontend is launched in a sandbox environment, where user can test the application and modify the code either with text prompts or with a visual code editor.

   *Note: user can choose to skip this step and start developing smart contract code without a frontend component.*

4. **Smart Contract Code Generation**: Using provided description, the AI generates the smart contract code for the application. Under the hood, the AI uses a database with default smart contracts, MCP module and LLM for fine-tuning the code for the specific application (we will discuss this in more detail in the upcoming sections). Smart contract is also launched in a sandbox environment using locally run blockchain environment, where user can test the application and modify the code either with text prompts or with a visual code editor.

5. **Testing**: Having frontend and smartcontract launched locally in the sandbox environment, user tests the application. User can also test the application in the live environment, where the smartcontract is deployed to the chosen blockchain Testnet or Devnet (frontend is still running locally in the sandbox environment).

6. **AI-based auditing**: User can request an AI-based auditing of the smartcontract code. The AI audits the smartcontract code (using several LLMs) for security vulnerabilities and performance issues. System will provide a report with the results of the audit. User can request a fix for the issues found in the audit if any.

7. **Deployment**: When user is satisfied with the application, it can be deployed to production environment.

**Limitations:**

- User must connect their wallet in order to use the platform.
- User can only deploy the application to the chain that is supported by the wallet (for example, if user connected Solana wallet, they can only deploy the application to Solana blockchain).
- In order to use the platform user should purchase messages using their wallet.
- Fronted is deployed to the supported hosting provider, which is chosen by the platform. If user wants to deploy it elsewhere, they can do so by using the provided code.

# 3 Technical Overview

## 3.1 AImpact Smart Contract Templates

In order to improve generation of the smart contract code, the AImpact platform uses a database of smart contract templates. Those templates are selected by AImpact MCP module (we will discuss this in more detail in the upcoming sections). LLM uses the selected template as a base for the smart contract code generation and fine-tunes it to the specific application.

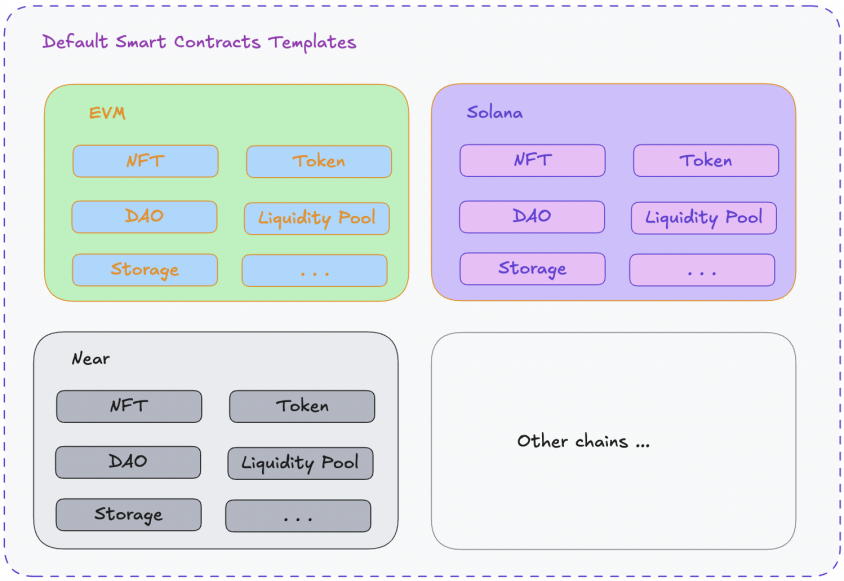


Figure 1: Smart contract templates

## 3.2 AImpact MCP Module

The platform uses AImpact MCP module to interact with the smart contract code generation and deployment. It provides LLM a set of tools to access the database of smart contract templates, deploy the smart contract either to the sandbox environment or to the chosen blockchain Testnet, Devnet or Mainnet, test the smart contract, deploy the frontend to the hosting provider.

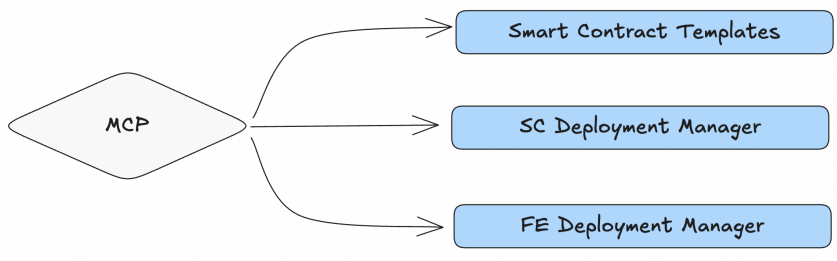


Figure 2: AImpact MCP Module